WHAT IS CLAIMED IS:

1. A method for a dynamic memory management with an object oriented program, the method comprising the steps of:

providing a memory block with a predetermined storage capacity; and

allocating objects to the memory block in sequence continuity.

2. The method according to claim 1, further comprising the step of providing a free reuse list to which an object released from the memory block is added as a reusable object corresponding to the released object.

3. The method according to claim 2, further comprising the steps of:

finding the reusable object in the free reuse list corresponding to an object requested for allocation;

allocating, when there is the reusable object corresponding to the requested object in the free reuse list, the requested object to a specific field of the memory block assigned to the reusable object and deleting the reusable object from the free reuse list; and

proceeding a new allocation subsequent to an object ranking on the latest order in the memory block when the free reuse list lacks the reusable object corresponding to the requested object.

4. The method according to one of claims 1 through 3, further comprising the steps of:

providing a free link list for object allocation;

assigning the memory block with the predetermined storage capacity larger at least than a threshold with reference to nodes of the free link list when there is an allocation request for a small object less than the threshold or the same with the threshold, the storage capacity of the memory block being enough to contain the requested small object for allocation; and

creating a node corresponding to a large object over the threshold with reference to the free link list when there is an allocation request for the large object.

5. A method of for a dynamic memory management with an object oriented program, the method comprising the steps of:

(a) providing a memory block for allocating a plurality of objects;

(b) allocating, when there is a reusable object corresponding to a requested object in a free reuse list, the requested object to a specific field of the memory block assigned to the reusable object and deleting the reusable object from the free reuse list;

(c) proceeding a new allocation subsequent to an object ranking on the latest order in the memory block when the free reuse list lacks the reusable object corresponding to the requested object; and

(d) adding an object released from the memory block to the free reuse list in correspondence with the released object..

6. The method according to claim 5, further comprising the step of comparing a size of the requested object with a predetermined threshold before providing the memory block, wherein objects smaller than the

threshold or the same with the threshold are managed by way of the steps (a) through (d) and objects larger than the threshold are managed by means of a memory management function.

7. The method according to one of claims 5 and 6, wherein the memory block is associated with a data structure, for an object allocated to the memory block, which includes information for the allocated object, an address pointer indicating an address to which the allocated object is assigned, and a released object pointer indicating an object released from the memory block; wherein the free reuse list is formed by an arrangement including elements of null pointers with reference to a predetermined size of the threshold; and wherein one of the null pointers which corresponds to the released object indicates the released object.

8. The method according to claim 7, wherein when the null pointer indicates a current one of the released object, the released object pointer assigned to the currently released object indicates a previous one of the released object.

9. A method for a dynamic memory management with an object oriented program, the method comprising the steps of:

forming a free link list for an object allocation;

allocating a large object over a predetermined threshold to a node which is made by coalescing a plurality of nodes to be same as a size of the large object with reference to the free link list;

sequentially allocating a small object, which is less than or the same

with the threshold, to a memory block having a storage capacity larger than the threshold; and

allocating, when there is a reusable object corresponding to the object in a free reuse list, the object to a specific field of the memory block assigned to the reusable object.

10. The method according to claims 9, wherein the memory block is associated with a data structure including information for the allocated object, an address pointer indicating an address to which the allocated object is assigned, and a released object pointer indicating an object released from the memory block; wherein the free reuse list is formed by an arrangement including elements of null pointers with reference to the threshold; and wherein one of the null pointers which corresponds to the released object indicates the released object.

11. The method according to claim 10, wherein when the null pointer indicates a current one of the released object, the released object pointer assigned to the currently released object indicates a previous one of the released object.

12. The method according to claim 9, further comprising the step of establishing a size of the threshold by means of analyzing objects frequently created and extinct in the object oriented program.

13. An apparatus operable with an object oriented program, comprising:

means configured to form a free link list for an object allocation;

means configured to allocate a large object over a predetermined threshold to a node which is made by coalescing a plurality of nodes to be same as a size of the large object with reference to the free link list;

means configured to sequentially allocate a small object, which is less than or the same with the threshold, to a memory block having a storage capacity larger than the threshold; and

means configured to allocate, when there is a reusable object corresponding to the object in a free reuse list, the object to a specific field of the memory block assigned to the reusable object.

14. The apparatus according to claims 13, wherein the free reuse list is formed by an arrangement including elements of null pointers with reference to the threshold; and wherein one of the null pointers which corresponds to the released object indicates the released object.